Simulating chemical systems

Goga, N.

Published in:
Journal of chemical sciences

DOI:
10.1007/s12039-013-0490-y

Publication date:
2013

J. Chem. Sci. Vol. 125, No. 5, September 2013, pp. 1285–1292.

Simulating chemical systems: MPI and GPU parallelization of novel SD algorithms

N GOGA[1,2]

[1]Groningen Biomolecular Sciences and Biotechnology Institute, Zernike Institute for Advanced Materials, University of Groningen, Nijenborgh 4, 9747 AG Groningen, The Netherlands

[2]Politehnica University of Groningen, Faculty of Engineering in Foreign Languages, Splaiul Independentei 313, Bucharest, Romania

e-mail: n.goga@rug.nl

MS received 21 March 2012; revised 9 April 2013; accepted 28 June 2013

Abstract. Molecular dynamics is used for simulating chemical systems with the goal of studying a large range of phenomena starting from cell structures to the design of new materials, drugs, etc. A very important component of molecular dynamics is the use of well-suited atomistic and molecular modelling of the chemical systems. This article presents the MPI and GPU-CUDA parallelization of novel stochastic-Langevin dynamics algorithm that is used in molecular dynamics for controlling the temperature of simulated systems. The research has been performed in the Molecular Dynamics Group of the University of Groningen and it is going to be included in the next version of Gromacs tool of molecular dynamics (www.gromacs.org). The Langevin algorithm implemented is original and is based on impulse application of friction and noise. Theoretical background, implementation, an efficiency discussion and relevant simulation results are presented in the different sections of this article. The simulations used Martini water. The parallelization of the algorithms was done in two versions: one in MPI using domain decomposition and another version was done in CUDA.

Keywords. Molecular dynamics; GPU; paralelisation; simulations.

1. Introduction

Molecular dynamics is used for simulating chemical systems with the goal of studying a large range of phenomena starting from cell structures to the design of new materials, drugs, etc. A very important component of molecular dynamics is the use of well-suited atomistic and molecular modelling of the chemical systems. The purpose of this article is to present the MPI and GPU-CUDA parallelization of a novel algorithm of stochastic dynamics.

Stochastic thermostats have the advantage above the global thermostats that they maintain the correct canonical distribution and show a robust first-order decay of temperature deviations towards the reference temperature. Global thermostats of the *weak-coupling* type[1] show a convenient first-order decay of temperature deviations, but do not maintain a canonical distribution; in fact, the resulting distribution is in between a canonical and a microcanonical distribution, depending on the coupling constant used.[2] Moreover, it is known that such thermostats may cause uneven distribution of kinetic energy among different collective degrees of freedom, resulting in overheating of collective degrees of freedom that are only weakly coupled to other degrees of freedom at the expense of cooling of these other degrees of freedom (the *flying ice-cube effect*).[3] Global thermostats of the *extended-system* type, such as the Nosé–Hoover thermostat,[4] will maintain a canonical distribution in configurational space, but show strong oscillatory behaviour of the temperature deviation as a result of the order of the differential equation related to the system's extension. The Nosé–Hoover *chain* thermostat,[5] using a cascade of thermostats, alleviates but does not solve this problem. See Berendsen[6] for a comparative discussion of thermostats.

The purpose of this article is to present a simple and efficient algorithm for stochastic dynamics, in particular for simple Langevin dynamics and its implementation in Gromacs and CUDA. The design of stochastic algorithms for molecular simulation has been an important subject of research in the 1980's.[7–10] The paper[10] describes a sophisticated algorithm that fully maintains the accuracy of the Verlet algorithm by integrating the stochastic term over the time step. This algorithm is still standard in both the GROMOS and Gromacs simulation packages. Traditional stochastic algorithms[8,10,11] that integrate the stochastic equations of motion over a time step become very complex, requiring the

sampling of two random variables from a bivariate distribution.

A different approach, pioneered by Peters[12] for the case of dissipative particle dynamics (DPD), leads to simpler and still correct algorithms. The principle is to consider the physical process as a sequence of a Hamiltonian evolution over one time step, followed by an *impulsive* action of friction and noise. The latter modifies the velocities without advancing the time. Thus the evolution in phase space is the approximate application of the Liouville operator over a time step, followed by a transformation defined by the impulsive friction and noise. If it can be proven that this impulsive action also leads to convergence to a canonical (i.e., Maxwellian) distribution at the reference temperature, this physical process is equally valid to achieve our goal of introducing an effective thermostat. The energy and momentum transfer implicit in the impulsive friction will influence transport properties in a controllable way. The behaviour as a thermostat will be robust, while the algorithm remains very straightforward and simple to implement. The main difference with the usual stochastic differential equation is that the time evolution of the system is not described by a single stochastic differential equation, but by a sequential application of a Hamiltonian evolution over a time step and an impulsive stochastic action on the velocities. Both steps should conserve the canonical distribution in phase space.

This approach is reminiscent of the principle of the Andersen thermostat,[13] which applies impulsive redistributions to particle velocities with a given probability Γh from a Maxwellian distribution. In the following, we extend Peters' DPD-type impulsive friction and noise (which applies relative velocity changes in the interparticle direction only) to Langevin dynamics.

This article is organized as follows. In section 2 the impulsive scheme for Langevin dynamics using the leap-frog algorithm is discussed. Section 3 describes the simulation details and reports the computational efficiency of the different methods; section 4 gives the results of various tests of the algorithms on Martini coarse-grained water. Both thermostat and diffusion behaviour are considered. Section 5 describes the implementation of the stochastic algorithm in CUDA. Discussion and conclusions are provided in section 6.

2. The impulsive Langevin leap-frog algorithm for systems without constraints

Consider a system of n particles with $3n$ degrees of freedom and consider every degree of freedom separately. Assume $v(t-\frac{1}{2}h)$, $x(t)$ and $F(t)=ma$ are the known velocity, coordinate and force component, and a the acceleration at time t of that degree of freedom. The impulsive Langevin extension of the leap-frog algorithm then reads as follows.

For all degrees of freedom do:

1. $v = v(t-\frac{1}{2}h) + ah$
2. $\Delta v = -fv + \sqrt{f(2-f)(k_B T_{\mathrm{ref}}/m)}\,\xi$
3. $x(t+h) = x(t) + (v + \frac{1}{2}\Delta v)h$
4. $v(t+\frac{1}{2}h) = v + \Delta v$.

Here, step 1 is the usual MD velocity-update of the leap-frog scheme, step 2 is the impulsive application of friction (reducing the velocity by a fraction $f : 0 \le f \le 1$) and noise (ξ is a random sample from a normal distribution). Step 3 updates the coordinates, taking into account that Δv is applied only between $t+\frac{1}{2}h$ and $t+h$: in fact, step 3 can be considered as two half steps (see figure 1):

3a. $x(t+\frac{1}{2}) = x(t) + v\frac{1}{2}h$,
3b. $x(t+h) = x(t+\frac{1}{2}h) + (v+\Delta v)\frac{1}{2}h$.

Step 4 assigns the modified velocity to the velocity at the end of the time step. It is irrelevant whether these steps are carried out sequentially per degree of freedom, sequentially per 3-D vector per particle, or each performed on a $3n$-D vector over all degrees of freedom.

The variance of the noise term is chosen such that the variance of the velocity

$$\langle (v+\Delta v)^2\rangle = (1-f)^2\langle v^2\rangle + f(2-f)\frac{k_{\mathrm{B}}T_{\mathrm{ref}}}{m}, \quad (1)$$

tends to the stationary value of $k_{\mathrm{B}}T_{\mathrm{ref}}/m$. This is easily seen from (1) by substituting $k_{\mathrm{B}}T_{\mathrm{ref}}/m$ for each of the mean squared velocities.

In this algorithm it is assumed that all degrees of freedom are subjected to the same friction and noise

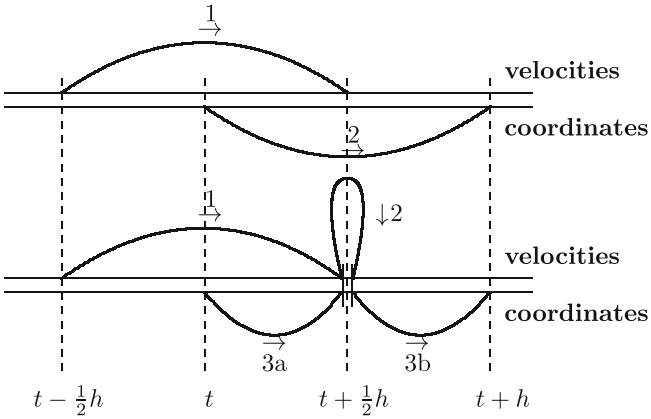

Figure 1. Traditional leap-frog scheme (top), and leap-frog scheme with impulsive phase (bottom).

at every time step; this is just a convenient scheme that could be replaced by other variants, e.g., different f's for different particles, or application of friction and noise to a (randomly) selected subset at every step. Note that the limiting case $f = 1$ completely removes the velocity and replaces it by a sample from a Maxwellian distribution. Thus, if the impulsive friction and noise is applied with $f = 1$ and with a probability Γh per particle per step, the Andersen thermostat is recovered. A *smoothed Andersen thermostat* with the same *average* velocity reduction factor will be obtained by applying the impulsive friction and noise with $f = \Gamma h$ at every step to every degree of freedom.

The algorithm is expected to be robust, in the sense that the impulsive term is exact, independent of the time step used. There is a lot of freedom of choice in the way the impulsive term is applied: the damping factor f may be applied to a random selection of particles and may differ for different particles.

It is obvious that the new velocity of any particle does not have the same direction and magnitude as its old velocity. As the random changes are uncorrelated, the total momentum is not conserved and neither is the total energy conserved. However, the average kinetic energy and hence the temperature will be stable: they tend toward the values determined by the reference temperature.

2.1 *Is the velocity distribution canonical?*

In order to judge the acceptability of the proposed procedure, we ask the following questions: *Assume the velocity distribution before the impulse is $\rho_0(v)$: (a) What will the distribution $\rho_1(v + \Delta v)$ be after the impulse? (b) What is the stationary distribution?*

After the impulse, the distribution $\rho_1(v + \Delta v)$ is the convolution of the original distribution $\rho_0(v)$ and the Gaussian distribution of the random term $\Delta v + fv$, which has a variance of $f(2 - f)k_B T_{ref}$:

$$\rho_1(w) = [2\pi f(2-f)k_B T_{ref}]^{-1/2} \times \int_{-\infty}^{\infty} dv\, \rho_0(v)\, \exp\left[-\frac{\{w-(1-f)v\}^2}{2f(2-f)k_B T_{ref}}\right], \quad (2)$$

where $w = v + \Delta v$.

This is the answer to question (a). The answer to question (b) is found by inserting the canonical distribution for v:

$$\rho_0(v) = [2\pi k_B T_{ref}]^{-1/2} \exp\left[-\frac{v^2}{2k_B T_{ref}}\right] \quad (3)$$

into (2). Carrying out the integration over v we find that $\rho_1(w)$ is exactly equal to the same canonical distribution ρ_0. So $\rho_0(v)$ is the stationary distribution. Thus, the impulsive application of friction and noise preserves not only the variance (as designed), but it preserves the complete canonical distribution.

2.2 *How does the temperature behave with time?*

A good thermostat should force a deviation from the reference temperature back to zero. How does the impulsive Langevin thermostat behave in this respect?

Consider one-dimension. The temperature is given by

$$T = \frac{m}{k_B}\langle v^2 \rangle. \quad (4)$$

The energy change ΔE_1 resulting from a single application of friction and noise to one degree of freedom is

$$\Delta E_1 = \frac{1}{2}m(v + \Delta v)^2 - \frac{1}{2}mv^2. \quad (5)$$

Using (1) and (4) this rewrites to

$$\Delta E_1 = \frac{1}{2}f(2 - f)k_B(T_{ref} - T); \quad (6)$$

the total energy change *per time step h* is the sum over all one-dimensional frictional events that occur per time step:

$$\Delta E_{tot} = \frac{1}{2}k_B \sum f(2 - f)(T_{ref} - T). \quad (7)$$

The energy change is initially supplied to the kinetic energy of the system, thus changing the temperature. However, when the rate of change is small, the energy change will be distributed over kinetic and potential energy. The temperature change is then determined by the total heat capacity C_V of the system:

$$\Delta T = \frac{\Delta E_{tot}}{C_V}, \quad (8)$$

yielding a differential equation for the time-dependence of the temperature

$$\frac{dT}{dt} = \frac{1}{2}k_B \frac{\sum f(2 - f)}{C_V h}(T_{ref} - T). \quad (9)$$

In the case of a three-dimensional application to N particles (9) has the form

$$\frac{dT}{dt} = \frac{3k_B}{2c_V}\frac{f(2 - f)}{h}(T_{ref} - T), \quad (10)$$

where

$$c_V = \frac{C_V}{N}, \quad (11)$$

is the specific heat per particle.

Equation (10) shows that any deviation from the reference temperature will decay to zero according to a first-order kinetic process

$$\frac{dT}{dt} = k_{\text{th}}(T_{\text{ref}} - T), \tag{12}$$

with rate constant

$$k_{\text{th}} = \frac{3k_{\text{B}}}{2c_V}\frac{f(2-f)}{h}. \tag{13}$$

Alternatively, the decay can be characterized by a time constant $\tau_T = 1/k_{\text{th}}$. Note that for an ideal gas $c_V = \frac{3}{2}k_{\text{B}}$, reducing the left fraction in (13) to 1; for atomic fluids this fraction is usually 2 to 3 times smaller.

The thermal rate constant can be expressed in an *effective friction rate* γ_{eff}, defined by the continuous friction rate that would reduce the velocity per time step h by a fraction f:

$$\gamma_{\text{eff}} \stackrel{\text{def}}{=} \frac{1}{h}\ln(1-f), \tag{14}$$

yielding

$$k_{\text{th}} = \frac{3k_{\text{B}}}{2c_V}\frac{[1-\exp(-2\gamma_{\text{eff}}h)]}{h} \approx \frac{3k_{\text{B}}}{2c_V}2\gamma_{\text{eff}}, \tag{15}$$

the latter value being a good approximation for small γh.

Thus, the thermostat is robust: the system temperature automatically decays to the reference temperature and the velocity distribution evolves into the proper canonical (Maxwellian) distribution.

2.2a *Slow and fast thermostats*: We note that this rate equation is valid when the constant rate of the thermostat is smaller than the rate of exchange between kinetic and potential energy of the system, which gives the system time to equilibrate between kinetic and potential degrees of freedom. Usually, this condition is fulfilled. If on the other hand, k_{th} is much larger ('fast' thermostats), C_V in (8) should be replaced by $E_{\text{kin}}/T = \frac{1}{2}k_{\text{B}}n_{\text{dof}}$, where n_{dof} is the number of degrees of freedom in the system. The equivalent of (9) for fast thermostats is

$$\frac{dT}{dt} = \frac{\sum f(2-f)}{n_{\text{dof}}h}(T_{\text{ref}} - T), \tag{16}$$

where the sum is taken over all one-dimensional frictional events that occur per time step. This means that for a fast thermostat c_V in equation (15) should be replaced by its ideal-gas value $3k_{\text{B}}/2$.

For the Langevin application, where a velocity reduction to $(1-f)v$ is applied to every degree of freedom at every time step, we defined an effective friction rate

$$\gamma_{\text{eff}} = -\frac{\ln(1-f)}{h}. \tag{17}$$

3. Computational details

3.1 *Simulation details*

The algorithm presented in the previous sections was implemented in the GROMACS program package[14] version 4.0.7, using parallelization based on domain decomposition.

One type of systems was used to test performance of the investigated coupling schemes: MARTINI[15] water. All simulations were performed in a periodic cubic box with dimensions longer than twice the cut-off distance. A cut-off distance for non-bonded interactions was set at 1.2 nm for MARTINI water systems. For MARTINI a potential shift function was applied,[16] with switch value of 0.9 to remove cut-off effects. The neighbour list was updated every step in order to remove any deviations due to computational errors. In all simulations a time step of 2 fs was used. For equilibration, weak pressure coupling[1] was applied with time constant of 2.0 ps and reference pressure of 1 bar; production runs were performed under constant volume conditions. The reference temperature was set to 320 K for MARTINI water system.

The MARTINI water system consisted of 3200 particles with mass 72 u in a cubic box of $(7.28856\,\text{nm})^3$. The initial velocities of the particles were obtained from a Maxwell–Boltzmann distribution corresponding to the chosen initial reference temperature. Simulations were either 10 ns or 50 ps long, for diffusion and thermal rate calculations, respectively.

For the computation of the diffusion coefficient, we used the mean square displacement (MSD) and applied the Einstein relation $D = \langle r^2(t)\rangle/(6t)$. The diffusion coefficient was calculated by least squares fitting a straight line through the MSD from 500 ps to 2 ns. Thermal rate constants were determined from least-squares fits to a single exponential of the temperature after switching the reference temperature at time $t = 0$ from 350 to 320 K. Each case was repeated 8 times, yielding 8 independent determinations k_i of the constant rate; we report the averages $\overline{k}$ with standard uncertainty σ computed from $\sigma^2 = \sum_{i=1}^{8}(k_i - \overline{k})^2/56$.

4. Results

4.1 *MARTINI coarse-grained water*

In the presence of an intrinsic diffusion coefficient, there is no reliable theory for the behaviour of the diffusion coefficient as a function of applied friction. One might naively suppose that internal and external friction would be additive, i.e., that the inverse diffusion

coefficient would be the sum of the inverse intrinsic diffusion coefficient and the inverse diffusion coefficient of the ideal gas. However, this assumption leads to a much higher diffusion coefficient than is actually observed. It turns out that the observed inverse diffusion coefficient obeys the following empirical linear relation to γ:

$$\frac{1}{D} = \frac{1}{D_{\text{intr}}} + C\gamma_{\text{eff}}, \tag{18}$$

where C is a proportionality constant with dimension $\text{ps}^2\,\text{nm}^{-2}$, which depends on the nature of the intrinsic interaction between the particles. The inverse diffusion coefficient is given in figure 2 and has the value $C = 250\ \text{ps}^2\,\text{nm}^{-2}$.

For prediction of the thermal rate the following properties of MARTINI water at 320 K are needed: $c_V = 0.0234(3)\,\text{kJ mol}^{-1}\,\text{K}^{-1}$, $\langle 1 - r/r_c\rangle = 0.24336$, $\langle (1 - r/r_c)^2\rangle = 0.089437$. For the Langevin case, the equation for the thermal rate constant k_{th} (see section 2.2) is for a fluid

$$k_{\text{th}} = \frac{3k_{\text{B}}}{2c_V}\frac{f(2-f)}{h} = \frac{3k_{\text{B}}}{2c_V}\frac{[1-\exp(-2\gamma h)]}{h}, \tag{19}$$

where $\gamma = -[\ln(1-f)]/h$. As MARTINI water is governed by an intrinsic potential, c_V is larger than its ideal-gas value of $3k_{\text{B}}/2$; in fact $3k_{\text{B}}/(2c_V) = 0.533$. Thus, we expect a 'slow' thermostat at low friction rates and an almost twice as large 'fast' thermostat at high friction rates (see section 2.2).

Figure 3 compares the theoretical prediction with the simulated thermal rate constants. The agreement with the theory is good. The full-drawn line gives the 'slow' behaviour and the broken red line indicates the 'fast' behaviour. The cross-over time between the two regimes occurs around $\gamma = 10$, or 0.1 ps, which is in the expected range.

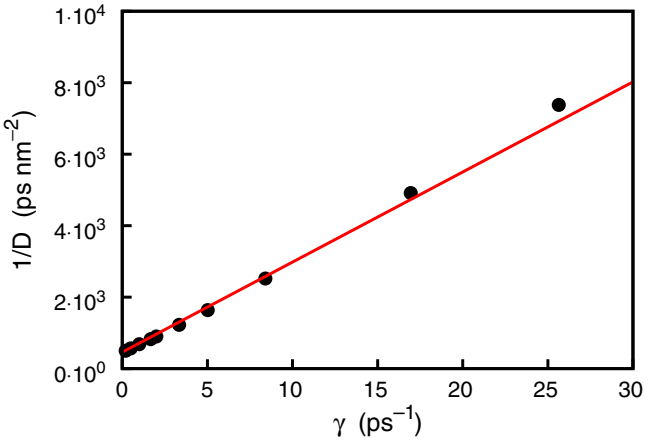

Figure 2. Inverse diffusion coefficient versus friction rate for MARTINI coarse grained water, simulated with impulsive Langevin algorithm. Black dots simulations, red line fit to $(D_{\text{intr}}^{-1} + C\gamma; C = 250\ \text{ps}^2\,\text{nm}^{-2})$.

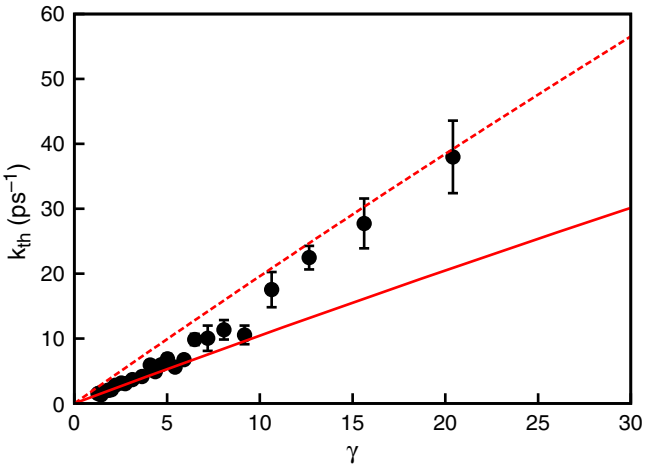

Figure 3. Thermal relaxation rate versus friction rate for MARTINI coarse grained water, simulated with impulsive Langevin algorithm. Black dots simulations; red lines theory, full-drawn for 'slow' and dotted for 'fast' thermostats.

We also did measurements for temperature. The average temperature for different friction coeficients differs very slightly as compared to reference temperature (320 K), the deviations being bellow 0.5 K.

4.2 *Performance*

We ran efficiency measurements on a workstation with Dual Core AMD Opteron™ 865 (1800 Mhz) processor. The comparison was done on a Martini water system with 3200 CG particles and with a time step of 10 fs, using Gromacs version 4.0.7. For Langevin dynamics we compared the 'new' impulsive algorithm with the 'old' algorithm available in Gromacs, which is based on integration of continuous friction,[10] and with pure MD without and with global thermostat. In the table below the 'speed' is reported as the number of nanoseconds simulated per day (24 h).

Method	ns/day
MD, no thermostat	69.90
MD, Berendsen thermostat	69.81
SD, Langevin, 'old'	59.79
SD, Langevin, 'new'	67.44

From these efficiency experiments, we can conclude that our new algorithm performs better than the existing algorithms from the literature or those implemented in Gromacs.

5. Implementation on CUDA

The Gromacs package is composed not only of an MD simulator, but it also has a variety of tools for analysing and visualizing the output of the simulations.[16] Its functionality is enabled by many lines of code and is dependent on the mathematical models implemented in it. It also employs a multitude of scientific algorithms and several dozen functions (called ńon-bonded kernels)́ for the short-range non-bonded interactions, each offering a different combination of methods for electrostatic and van der Waals forces.

The OpenMM library[17] is developed by a team from Stanford University, that Gromacs interfaces in order to be able to run simulations on GPU as well. The OpenMM library has support both for CUDA architecture provided by NVIDIA and OpenCL, the open standard defined by Khronos Group. It has a quite wide range of algorithms implemented that run on GPU, but still it is not a valid equivalent of Gromacs: its functionality is lower as compared to Gromacs. This is explained by the fact that OpenMM has a recent development history while Gromacs has a much longer development history. In the combination of Gromacs/OpenMM our new algorithms were developed.

5.1 *Algorithms implementation*

We developed an implementation of the new SD thermostat for the CUDA architecture. The main computational flow is represented in figure 4 while in figure 5 we represent the GPU parallelization information flow.

First the integrator object is created. After creating it, its corresponding function will be executed. The functions that use the integrators are written in plain C, but they call methods and functions that use CUDA, implicitly the kernel functions. Once a kernel is launched, it is executed on GPU.

In consequence, thousands of threads are created, prepared to execute the same piece of code that resides in the kernel. Once the kernel is prepared to be executed, the data from structures that are called inside the kernel code are brought in the memory of the GPU and also the relevant data, in the memory of each multiple processors. This way, the threads have direct access to the data relevant to them. Once all the threads are finished and the kernel code has been executed, the results are sent to the processor and RAM, and the execution is continued on the CPU.

In order to use this integrator, the friction factor and the reference temperature must be specified in the *.mdp file. In the openmm_wrapper.cpp file of Gromacs, when this integrator is called, a new object from the SDNew Integrator is instantiated. This way, the specific methods and kernels are called. The main algorithm is located in the kSDNewUpdate.h. The kernel methods are called in the file CudaKernels.cpp, where the SDNew integrator is created and used.

5.2 *Performance results*

The SD thermostat was tested on the same system of Martini coarse grained water. The number of particles varied from small systems to bigger systems. All the systems were tested on a quadcore machine with a processor type Intel (R) Core (TM) i7 with 2.67 Mhz frequency and 1.5 Gb internal memory per core and a NVIDIA GEForce 9600 GT graphical card with 512 Mb memory and 64 GPU processors. When comparing the new algorithm with the old SD algorithms, the difference in performance was not that visible as in the previous case due to the overhead introduced by the communication between main memory and NVIDIA CARD (see below) which makes the performance improvement not to be visible.

Comparing the ns/day obtained for one processor, for two processors and on NVIDIA, it can be noticed that the performance on NVIDIA is increased with almost 70% comparing to one processor for the same machine. But this happens on systems, with less than 80000 particles. For larger systems the performances on NVIDIA card start to be comparable to just one processor (see figure 6). Because of the reduced memory of

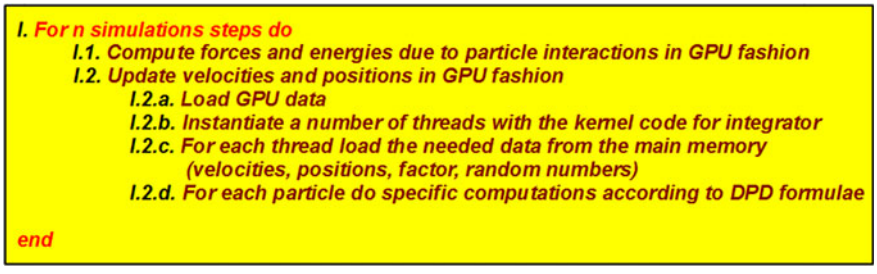

Figure 4. Main computational flow.

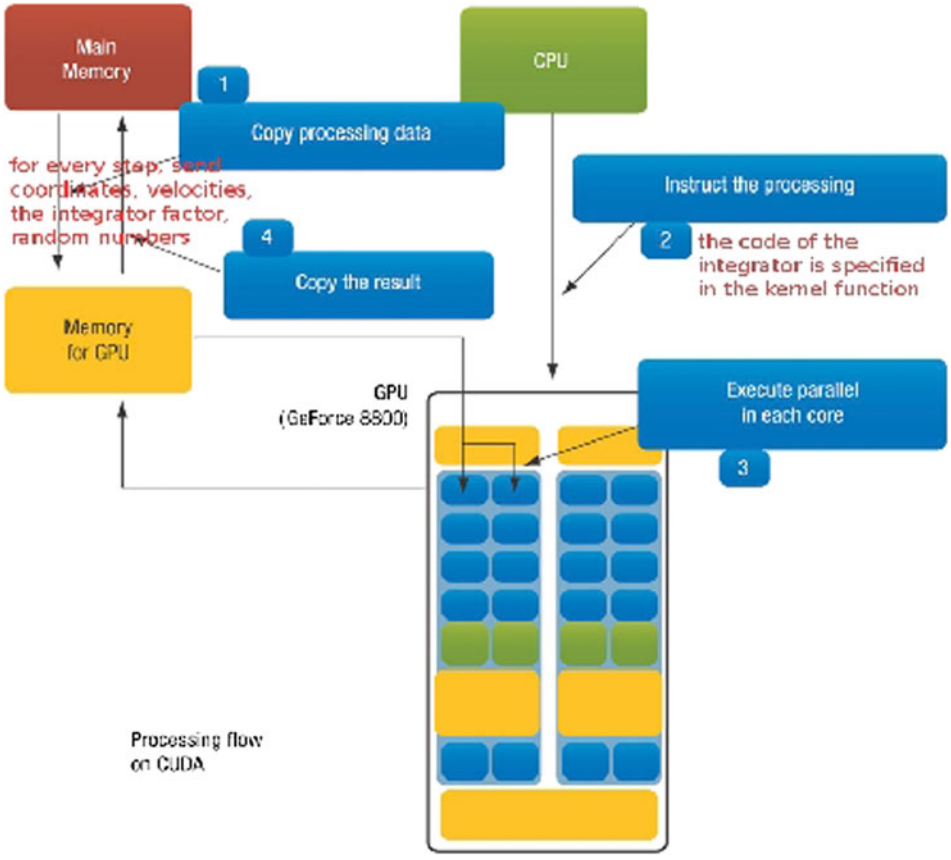

Figure 5. Main information flow on CUDA.

the GPU, a big number of particles cannot be loaded on the board. Also it means that the interchange of data between the GPU shared memory of each microprocessor and the main slow memory of the GPU occurs very often, which increases the latency. For larger systems there it is needed more data transfers between the GPU memory and the ram memory, fact that explains the decrease in the performance of the algorithm.

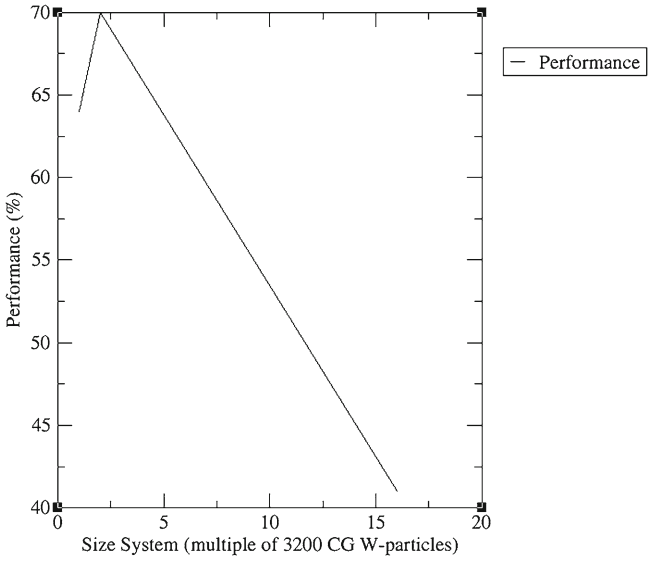

Figure 6. Performance results.

6. Discussion and conclusions

In this article, we have shown that the application of impulsive friction and noise, as introduced by Peters,[12] provides a valid implementation for Markovian stochastic dynamics with predictable thermostat behaviour. This is true for the traditional particle-based simple Langevin dynamics.

What are the advantages and disadvantages of the application of impulsive friction and noise? Firstly, the phase space distribution remains canonical, which cannot be guaranteed for weak-coupling global thermostats.[1] Secondly, the temperature response is a smooth first-order decay, which avoids many problems of extended-system global thermostats.[4,6]

We also described the MPI and the GPU parallelization of novel SD of molecular systems simulations. The new algorithms were developed by the Molecular Dynamics Group of the University of Groningen. The

algorithms were parallelized on the Gromacs/OpenMM software for molecular dynamics. The new SD algorithm is going to be introduced in the next release of Gromacs tool of molecular dynamics. For MPI efficiency experiments, we can conclude that our new algorithm performs better than the existing algorithms from the literature or implemented in Gromacs.

For CUDA, the performances of the new SD algorithms are similar with the previous SD algorithms implemented in Gromacs because of the overhead produced by the communication between internal memory and Nvidia card. The performances of the algorithm have also been compared with the performances of the code on one, two and four processors. It can be observed that the performance obtained on the GPU is greater than on a single processor, but not equivalent with two processors. Comparing the ns/day obtained for one processor, for two processors and on NVIDIA, it can be noticed that the performance on NVIDIA is increased with almost 70% compared to one processor. This happens on systems with less than 80000 particles. For larger systems the performances on NVIDIA card start to be comparable to just one processor. Similar conclusions for larger atomistic systems were drawn for other MD algorithms on GPU in literature (see for example ref 18). It can be concluded that the parallelization through the use of graphical cards improves the performances of the runs as compared to the serial version of the code (for the case of atomistic systems with smaller number of particles).

Based on the observed results, for the case of systems with larger number of particles, the performances on GPU are not as good as for the ones with smaller number of particles. Therefore there is room for more improvements, for designing and implementing better computational algorithms (in future) that will improve the GPU performances for the case of systems with larger number of particles (larger than 80000 particles). This can be a direction for future work.

The presented algorithms are going to be included in the new Gromacs tool release. As applications of the novel SD algorithms presented, this model is going to be used for studying a large range of chemical systems for a large range of phenomena.

References

1. Berendsen H J C, Postma J P M, van Gunsteren W F, Dinola A and Haak J R 1984 *J. Chem. Phys.* **81** 3684
2. Morishita T 2000 *J. Chem. Phys.* **113** 2976
3. Harvey S C, Tan R K Z and Cheatham T E 1998 *J. Comput. Chem.* **19** 726
4. Hoover W G 1985 *Phys. Rev.* **A31** 1696
5. Martyna G J, Tuckerman M E and Klein M L 1992 *J. Chem. Phys.* **97** 2635
6. Berendsen H J C 2007 *Simulating the physical world, A hierarchy of models for simulation* (Cambridge, UK: Cambridge University Press)
7. Van Gunsteren W F, Berendsen H J C and Rullmann J A C 1981 *Mol. Phys.* **44** 69
8. Van Gunsteren W F and Berendsen H J C 1982 *Mol. Phys.* **45** 637
9. Van Gunsteren W F and Berendsen H J C 1983 in *The physics of superionic conductors and electrode materials* (ed) J W Perram, *NATO ASI Series* **B92** (New York: Plenum) p. 241
10. Van Gunsteren W F and Berendsen H J C 1988 *Mol. Simul.* **1** 173
11. Allen M P 1980 *Mol. Phys.* **40** 1073
12. Peters E A F J 2004 *Europhys. Lett.* **66** 311
13. Andersen H C 1980 *J. Chem. Phys.* **72** 2384
14. Hess B, Kutzner C, van der Spoel D and Lindahl E 2008 *J. Chem. Theory Comput.* **4** 435
15. Marrink S J, Risselada H J, Yefimov S, Tieleman D P and de Vries A H 2007 *J. Phys. Chem. B* **27** 7812
16. van der Spoel D, Lindahl E, Hess B, van Buuren A R, Apol E, Meulenhoff P J, Tieleman D P, Sijbers A L T M, Feenstra K A, van Drunen R and Berendsen H J C 2005 *Gromacs User Manual version 4.0*, www.gromacs.org
17. OpenMM, https://simtk.org/home/openmm
18. Friedrichs S, Eastman P, Vaidyanathan V, Houston M, LeGrand S, Beberg A L, Ensign D, Bruns C M and Pande V S 2009 *J. Comput. Chem.* **30(6)** 864872